DATA ANALYSIS AND VISUALISATIONS OF INDIAN PREMIER LEAGUE

Nithin Kankanti
Computer Science
University of Colorado
Boulder Boulder Colorado
United States
nika9944@colorado.edu

Mohan Dwarampudi
Computer Science
University of Colorado
Boulder Boulder Colorado
modw1211@colorado.edu

Sankaranarayanan
Computer Science
University of Colorado
Boulder Boulder Colorado
United States
sana6469@colorado.edu

INTRODUCTION

The Indian Premier League is a professional Twenty20 cricket league in India contested during March or April and May of every year by eight teams representing eight different cities in India. In this project, we want to present some cool, interesting fun facts, records and results of the Indian Premier League.

Data is a very important component in doing data mining/analysis. When there is structured data available, it is all the more easier to do logical and statistical analysis on the data. The data can either be stored in a data warehouse or can be obtained from an API on a periodic basis.

Given the amount of data collected over the years, this is a perfect scenario to do data analysis, mining. Due to the abundance of data, it enables us to explore the various aspects of the data pipeline as well and how it can scale. This can be used in building a distributed data analysis/mining system where we can explore and experiment with the architecture of the underlying system.

A huge amount of data indicates that Data Visualizations can also be utilized to generate useful insights. It can be used as a player's feedback which will help them in improving the performance.

KEYWORDS

Data Visualization, Distributed Data Analysis, Data Mining

RELATED WORK

Previous work in this domain involved doing Fuzzy Clustering on IPL data for data mining.Fuzzy clustering is an extension of the cluster analysis, which represents the affiliation of data points to clusters by memberships. Introducing fuzziness to clustering gives us the flexible representations of substructures of the data set.There are different shapes of cluster centers and prototypes. Most of them conduct clustering in accordance with similarity or dissimilarity derived from distances from the centroid of the cluster of the data points.

Other work in this domain involved doing Predictive Analysis on the dataset. Examples include analysis on the dependency between winning the toss and batting and whether the decision was successful in the end. It also includes trivial analysis on how strategies can be implemented given the pitch and weather conditions. Some examples include analysis of players in the teams and the correlation between the type of pitch and strengths of the players in the team.

Moreover, the possibility of a simulation has also been explored. This involves artificially creating an environment with the conditions, players and taking into consideration the strengths and weaknesses of the players and running a simulation. This way, variables can be tweaked to observe the performance and the strategies can be varied accordingly.

These are some of the related works in the past in this domain. Most of these works involved building a feedback mechanism to support the players.

PROPOSED WORK

In this project we are going to build an end to end ETL pipeline to mine the data to extract valuable insights for better visualizations. ETL Processing will be done via amazon web services data pipeline due to the amount of input data.
The ETL Pipeline that we came up with is as follows:

EXTRACT:

- Data has been sourced from multiple areas -
  - Scrapping from popular cricketing websites.
  - Scrapping wiki pages.
  - Through Google API for Geo points.
  - Kaggle datasets.
- All the data is then stored in Amazon S3, which is then pushed into DynamoDB. S3 event invokes AWS Lambda which does the data parsing before it is rested in DynamoDB.
- The entire data has been utilized into three tables in DynamoDB, namely - deliveries, matches, players. This data acts as a source of truth for all the further operations.

TRANSFORM:

- All the semi-parsed data is transformed into a meaningful entry - JSON.
- Triggers on DynamoDB would invoke AWS Lambda whenever a new entry is added into DynamoDB.
- AWS Lambda transforms the data into meaningful patterns, which are further loaded into the ElasticSearch cluster.
- AWS Lambda also fetches additional Geo data through Google API.
- AWS Lambda uses Redis for a quick Key: Value mapping lookup.

LOAD:

- ElasticSearch indices all the incoming data from Lambdas.
- Data on ElasticSearch is split on the nodes in the cluster.
- All the 3 formats of the data are stored in different indices -
  - deliveries
  - matches
  - players

ELASTIC SEARCH:

- A two node cluster, served out via Load Balancer.
- Load Balancer endpoint would be the face of ElasticSearch.
- There are separate indices for all three types of data sources which are mentioned before.

KIBANA:

- This is the face of our project.
- Dashboards are generated separately for each index.
- Visualizations such as Geo-tagging, Heatmaps, Custom Metrics, Pie Charts, Bar Graphs, Line Plots, etc. are integrated.
- Filters applied on a visualization is applied across the dashboard.
- Also Kibana is useful in visualizing large datasets and reduces the amount of manual effort in selecting the attributes.

NGINX

- Utilized for the purpose of port forwarding.

- It forwards the request received on 80 to the Kibana's listening port, making Kibana as the face of the application

INITIAL WORK:

We started out by doing data extraction from sources such as Kaggle and doing preliminary data pre-processing. After this we decided to do initial analysis of data by using a few important statistically important measures like the Pearson Correlation Coefficient. This was useful in identifying the variables that are positively/negatively correlated and helped in filtering out a few columns. But this measure only works for numerical attributes. So we decided to do basic visualizations using the Python library called Seaborn. This was used in preparing a scatter plot of a pair of variables and also a heatmap of non-numerical attributes. This gave us an idea about non-numerical attributes as well. But generating such visualizations eventually becomes a difficult task especially when the data is huge and there is a lot of manual work involved. To reduce the amount of manual analysis and to spend time effectively in generating the insights, we decided to move the visualizations over to Kibana.

From the Machine Learning perspective, we decided to aggregate the data over all the seasons. This was useful in performing a preliminary K-Means analysis. Different performance metrics were used for analysis. Batsmen were classified into high performing, moderately performing players based on the number of runs scored across all the seasons. Bowlers were classified based on the number of wickets. A lot of such performance metrics can be used but could become a little tedious on large datasets. We had a large dataset that consisted of 150000 records and needed to interface with the other datasets as well. So we decided to move to ElasticSearch and Kibana which help in this purpose.

EVALUATION:

- We will be using kibana to generate dashboards separately for each of the indexes.
- We can apply filters on a visualisation to come up with the required metrics.
- We will be generating different visualizations such as Geo Tagging, Heat Maps, Custom Metrics, Pie charts.
- Our project end product will be looking similar to the one mentioned below.

MILESTONES:

- **Data Extraction from different sources and storing into S3.**
- **Transforming the data into JSON Format.**
- **Creating AWS lambda functions to parse the data before loading into DynamoDB.**
- **Fetching geo data through Google API.**
- **Transformed data is loaded into an elastic search cluster.**
- **Indexing the incoming data from lambdas with elastic search.**
- **Loading the data into three different tables namely Matches, players and Deliveries.**
- **Generate visualisations separately for each of the tables with the created indices using Kibana.**
- **Create the dashboards with the already generated visualisations.**
- **Use NGINX to host the created dashboards.**

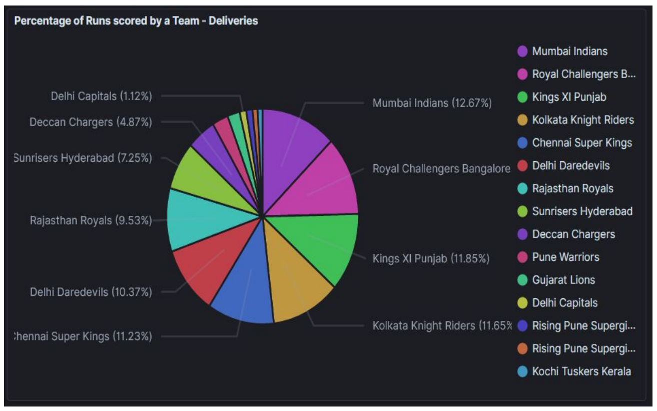
ISSUES ENCOUNTERED:

As the data was big, we had to ensure our data was properly indexed in ElasticSearch. Using DynamoDB to upload data to ElasticSearch took a while especially since one of our tables had 150000 records. So we had to resort to a bulk upload. Bulk upload requires the data to be a set of json objects with each object having the proper properties(index and type).

Maps Visualization was done on Kibana which required the use of GeoPoint Data Type.Kibana could not render the location if the latitude and longitude were Decimal Points.

One more common issue was that our indices were getting deleted on ElasticSearch. The exact reason was not known but based on a quick search it was identified that databases on the public internet were prone to a 'Meow' attack. So we had to secure the data by specifying proper security groups.

Sample Visualization:


Percentage of Runs scored by a Team - Deliveries
Delhi Capitals (1.12%)
Deccan Chargers (4.87%)
Sunrisers Hyderabad (7.25%)
Rajasthan Royals (9.53%)
Delhi Daredevils (10.37%)
Chennai Super Kings (11.23%)
Mumbai Indians (12.67%)
Royal Challengers Bangalore
Kings XI Punjab (11.85%)
Kolkata Knight Riders (11.65%
● Mumbai Indians
● Royal Challengers B...
● Kings XI Punjab
● Kolkata Knight Riders
● Chennai Super Kings
● Delhi Daredevils
● Rajasthan Royals
● Sunrisers Hyderabad
● Deccan Chargers
● Pune Warriors
● Gujarat Lions
● Delhi Capitals
● Rising Pune Supergi...
● Rising Pune Supergi...
● Kochi Tuskers Kerala